Commun. Comput. Phys.
doi: 10.4208/cicp.OA-2016-0164

Vol. **22**, No. 4, pp. 913-934
October 2017

A Moving-Least-Square Immersed Boundary Method for Rigid and Deformable Boundaries in Viscous Flow

Duc-Vinh Le[1,*] and Boo-Cheong Khoo[2]

[1] *Institute of High Performance Computing, A*STAR, 1 Fusionopolis Way, #16-16 Connexis, Singapore 138632.*
[2] *Department of Mechanical Engineering, National University of Singapore, Kent Ridge Crescent, Singapore 119260.*

Received 26 September 2016; Accepted (in revised version) 2 March 2017

Abstract. We present a moving-least-square immersed boundary method for solving viscous incompressible flow involving deformable and rigid boundaries on a uniform Cartesian grid. For rigid boundaries, noslip conditions at the rigid interfaces are enforced using the immersed-boundary direct-forcing method. We propose a reconstruction approach that utilizes moving least squares (MLS) method to reconstruct the velocity at the forcing points in the vicinity of the rigid boundaries. For deformable boundaries, MLS method is employed to construct the interpolation and distribution operators for the immersed boundary points in the vicinity of the rigid boundaries instead of using discrete delta functions. The MLS approach allows us to avoid distributing the Lagrangian forces into the solid domains as well as to avoid using the velocity of points inside the solid domains to compute the velocity of the deformable boundaries. The present numerical technique has been validated by several examples including a Poiseuille flow in a tube, deformations of elastic capsules in shear flow and dynamics of red-blood cell in microfluidic devices.

AMS subject classifications: 65M06, 76D05, 74F10, 92C10

Key words: Moving least squares, immersed boundary method, direct-forcing method, Navier-Stokes equations, cell sorting device.

1 Introduction

Viscous incompressible flows involving deformable boundaries and complex geometries are encountered in many engineering applications. In recent years, many Cartesian-grid methods have been developed for solving these types of problems to avoid the need of

*Corresponding author. *Email addresses:* ledv@ihpc.a-star.edu.sg (D.-V. Le), mpekbc@nus.edu.sg (B.-C. Khoo)

 ©2017 Global-Science Press

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

a re-meshing process. Among these developments, the Immersed Boundary (IB) method has shown great promises in handling flows with moving boundaries. The IB method was originally proposed by Peskin to study the blood dynamics in human heart [1, 2]. The IB method was further developed for many flow problems involving deformable boundaries [3–6] as well as rigid boundaries [7–9]. Lai and Peskin [7] used a stiff-spring forcing formulation to enforce the noslip condition at the solid boundaries. The stiff-spring approach requires a small time-step to preserve the stability. To avoid using small time-step, implicit forcing schemes [9, 10] were proposed to enforce the noslip boundary condition. An alternative approach to improve the time-step is the direct forcing formulation approach employed in [11–17] where the velocity boundary condition is imposed through an interpolation procedure. In the direct forcing method, forcing terms are enforced at the grid points next to the rigid boundaries. In other similar direct forcing approaches [8, 18], the forcing terms are applied directly at the positions of the rigid boundaries and are computed based on the momentum equations and the desired velocity at the rigid boundaries.

Other Cartesian-grid methods for complex geometries include cut-cell finite-volume approach [19–21], ghost-cell approach [22] and curvilinear IB method [23–25]. In the cut-cell approach, grid cells are re-shaped according to the local geometry if they are cut by the rigid boundaries. The gradients and fluxes across the faces of these cut cells are approximated using polynomial interpolating function to preserve second-order accuracy. Most applications of the cut-cell approach focused on 2D problems because the complex cut-cell procedure was not straightforward in three dimensions. In the ghost-cell method, ghost cells are defined as solid cells that have neighbor cells in the fluid domain and their ghost-values are extrapolated to enforce implicitly the noslip condition at the boundary. In the curvilinear IB approach [23–25], boundary conditions are reconstructed at grid nodes of a background curvilinear grid generated to closely approximate the shape of an arbitrarily complex solid boundary. More details of these methods can be found in [26, 27].

In the present work, we propose an IB method for solving the incompressible Navier-Stokes equations for viscous flow involving both rigid and deformable boundaries. Our approach for handling rigid boundaries is largely based on the direct forcing method [11–17] in which noslip boundary condition is taken into account implicitly by reconstructing the velocities at the fluid nodes in the immediate neighbor of the solid boundaries. In the previous direct-forcing method [11–17], the reconstructed velocity is calculated by linear interpolation along specific directions such as grid lines or the normal directions. Here, we propose a reconstruction approach that utilizes Moving Least Squares (MLS) method [28] to reconstruct the velocity near the solid boundaries. MLS method allows us to include information from all spatial directions and from more points in both fluid and rigid boundaries in the reconstruction procedure. The MLS reconstruction has been used in the context of embedded-boundary formulations to handle rigid boundaries [29, 30]. In the present study, we also use MLS interpolation procedure together with the IB method to handle deformable boundaries in the presence of solid boundaries. In the

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

original IB method, the force, torque and power are converted correctly between Eulerian and Lagrangian meshes through discrete delta functions [2]. The discrete delta function constructed on uniform grid [2] has the support domain of 4 meshwidths in each spatial direction. When the deformable boundaries are close to the rigid boundaries, the support domain of the discrete delta function may contain grid points inside the rigid region. To avoid distributing Lagrangian forces into the rigid regions, we propose using the MLS method to construct the distribution/interpolation shape functions instead of using the discrete delta functions. And to maintain the efficiency of the IB method, the MLS shape functions are only applied at some IB nodes closed to the solid boundaries while the discrete delta functions are still applied at the IB points whose interpolation stencils lie entirely within the fluid.

The next parts of the present paper are organized as follow. We begin with the governing equations and basic solver of the IB algorithm in Section 2. We then show the numerical treatment of rigid and deformable boundaries together with a brief description of the numerical implementation. We present some numerical examples in Section 3 and the conclusions in Section 4.

2 Numerical methods

2.1 Governing equations and projection method

In this work, we introduce a numerical method for handling viscous fluid flows with deformable boundaries and solid bodies immersed in the fluid. A typical interface problem that we consider is shown in Fig. 1 where a set of deformable interfaces Γ such as cells and elastic capsules interacts with the flow and solid bodies within the domain Ω. In this coupled system, the fluid and the immersed boundaries are governed by different set of partial differential equations. The fluid is governed by the Navier-Stokes (N-S) equations, written as

$$\rho\left(\frac{\partial \boldsymbol{u}}{\partial t}+\boldsymbol{u}\cdot\nabla\boldsymbol{u}\right)=-\nabla p+\mu\nabla^2\boldsymbol{u}+\boldsymbol{f}+\rho\hat{\boldsymbol{f}}, \tag{2.1}$$

$$\nabla\cdot\boldsymbol{u}=0, \tag{2.2}$$

where ρ, μ, p and $\boldsymbol{u}=(u,v,w)^T$ are the fluid density, viscosity, pressure and velocity, respectively. An external body force field $\hat{\boldsymbol{f}}$ is a fictitious force used to take into account the presence of the rigid boundaries through enforcing the noslip condition. A singular force $\boldsymbol{f}$ is the elastic force the deformable boundaries exert on the fluid and is written as

$$\boldsymbol{f}(\boldsymbol{x},t)=\int_{\Gamma(t)}\boldsymbol{F}(\xi^1,\xi^2,t)\delta(\boldsymbol{x}-\boldsymbol{X}(\xi^1,\xi^2,t))\mathrm{d}\Gamma, \tag{2.3}$$

where $\boldsymbol{F}(\xi^1,\xi^2,t)$ is the elastic force per unit area at a material point $\boldsymbol{X}(\xi^1,\xi^2,t)$ on the deformable interface $\Gamma(t)$ at time t. In (2.3), (ξ^1,ξ^2) is the coordinate defined on $\Gamma(t)$, $\boldsymbol{x}$ is a

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

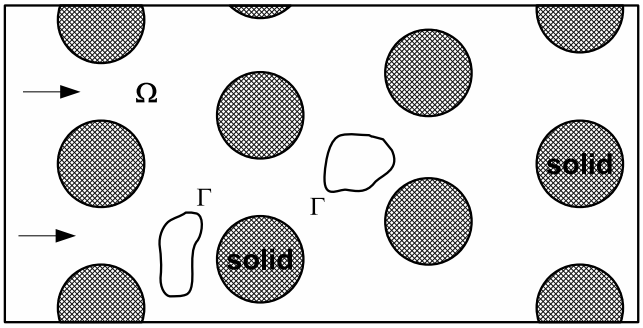


Figure 1: An example of an immersed boundary problem where deformable interfaces Γ interact with solid bodies and the flow within a rectangular domain Ω.

position in the fluid domain and $\delta(\boldsymbol{x})$ is a three-dimensional delta function $\delta(x_1)\delta(x_2)\delta(x_3)$, where $\boldsymbol{x}=(x_1,x_2,x_3)$. The deformable boundary moves with the fluid velocity as

$$\frac{\mathrm{d}\boldsymbol{X}}{\mathrm{d}t}=\int_\Omega \boldsymbol{u}(\boldsymbol{x},t)\delta(\boldsymbol{x}-\boldsymbol{X})\mathrm{d}\boldsymbol{x}. \tag{2.4}$$

The Navier-Stokes equations are discretized using a projection method [31] in which finite differences are employed on a standard marker-and-cell (MAC) staggered grid. Given the current $\{\boldsymbol{u}^n,p^{n-1/2}\}$ and the elastic force $\boldsymbol{f}^{n+1/2}$, we solve for the new $\{\boldsymbol{u}^{n+1},p^{n+1/2}\}$ in two steps:

1) Solve the Helmholtz equations for a tentative velocity $\boldsymbol{u}^*$:

$$\frac{\boldsymbol{u}^*-\boldsymbol{u}^n}{\triangle t}=-(\boldsymbol{u}\cdot\nabla\boldsymbol{u})^{n+\frac{1}{2}}+\frac{1}{\rho}\left(\frac{\mu}{2}\left(\nabla^2\boldsymbol{u}^*+\nabla^2\boldsymbol{u}^n\right)-\nabla p^{n-\frac{1}{2}}+\boldsymbol{f}^{n+\frac{1}{2}}\right)+\hat{\boldsymbol{f}}^{n+\frac{1}{2}}, \tag{2.5}$$

where the convection term is extrapolated using Adams-Bashforth formula, $(\boldsymbol{u}\cdot\nabla\boldsymbol{u})^{n+\frac{1}{2}}=\frac{3}{2}(\boldsymbol{u}\cdot\nabla\boldsymbol{u})^{n-\frac{1}{2}}-\frac{1}{2}(\boldsymbol{u}\cdot\nabla\boldsymbol{u})^{n-\frac{3}{2}}$ and the viscous term is calculated using Crank-Nicolson method.

2) Solve the Poisson equation and update the fluid velocity and pressure:

$$\nabla^2\phi^{n+1}=\rho\frac{\nabla\cdot\boldsymbol{u}^*}{\triangle t}, \tag{2.6}$$

$$\boldsymbol{u}^{n+1}=\boldsymbol{u}^*-\frac{1}{\rho}\Delta t\nabla\phi^{n+1}, \tag{2.7}$$

$$p^{n+\frac{1}{2}}=p^{n-\frac{1}{2}}+\phi^{n+1}-\frac{\mu}{2\rho}\nabla\cdot\boldsymbol{u}^*. \tag{2.8}$$

Eqs. (2.5) and (2.6) are solved using Fast Fourier Transform algorithms [32, 33]. The forcing terms $\hat{\boldsymbol{f}}^{n+\frac{1}{2}}$ and $\boldsymbol{f}^{n+\frac{1}{2}}$ in Eq. (2.5) must be computed in advance to enforce the noslip conditions at the rigid interfaces and to take into account the effect of the elastic boundaries, respectively. This will be discussed in the next two subsections.

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

2.2 Treatment of the rigid immersed boundaries

Our approach for handling rigid boundaries is largely based on Balaras's direct forcing method and its extensions [13–17]. In this direct-forcing approach, the forcing term $\hat{f}^{n+\frac{1}{2}}$ is calculated based on the discretized momentum equations as follows

$$\frac{\boldsymbol{u}^*-\boldsymbol{u}^n}{\triangle t}=\mathbf{rhs}+\hat{f}^{n+1/2}, \tag{2.9}$$

where the term **rhs** contains the advection, viscous, pressure gradient and elastic force terms. If an Eulerian background grid node coincides with the rigid boundary, the force $\hat{f}$ at this node can be computed as

$$\hat{f}^{n+1/2}=\frac{\boldsymbol{u}_\Psi-\boldsymbol{u}^n}{\triangle t}-\mathbf{rhs}, \tag{2.10}$$

where $\boldsymbol{u}_\Psi$ is the known velocity of the rigid boundary at the considered Eulerian background grid point. However, the grid node normally does not locate exactly on the rigid boundary Ψ, therefore a velocity interpolation procedure needs to be constructed at that node such that the desired velocity at the rigid boundary is enforced. In the direct-forcing method, the forcing term $\hat{f}^{n+1/2}$ is applied at the solid and forcing points and is zero at the fluid points. The forcing points are defined as the velocity grid points in fluid zone with at least one of their neighbors in the solid domain as shown as triangular symbols in Fig. 2(a). The solid points are velocity grid points in solid region and the fluid points are the remaining points in the fluid region. We also split pressure grid points into three different categories: (i) solid points, which are all the points in solid region; (ii) fluid points, which are pressure points in the fluid region and have at least one velocity neighbor which is a fluid point; (iii) pseudo-fluid points, which are the remaining pressure points in the fluid region.

In 2D, the force at a forcing point (i,j) is calculated as

$$\hat{\boldsymbol{f}}_{ij}^{n+1/2}=\frac{\boldsymbol{V}_{ij}^{n+1}-\boldsymbol{u}_{ij}^n}{\triangle t}+\frac{3}{2}H(\boldsymbol{u}_{ij}^n)-\frac{1}{2}H(\boldsymbol{u}_{ij}^{n-1})+\frac{1}{\rho}\left(\nabla p_{ij}^{n-\frac{1}{2}}-\boldsymbol{f}_{ij}^{n+\frac{1}{2}}\right), \tag{2.11}$$

where $\boldsymbol{V}_{ij}^{n+1}$ is the reconstructed velocity at the forcing point (i,j) such that the noslip condition is satisfied on the rigid boundaries and $H(\boldsymbol{u})$ is defined as $H(\boldsymbol{u})=\boldsymbol{u}\cdot\nabla\boldsymbol{u}-\frac{\mu}{\rho}\nabla^2\boldsymbol{u}$. For solid points, $\boldsymbol{V}_{ij}^{n+1}$ is set to the solid body velocity which is zero for stationary solid boundary. The extension of the forcing term in 3D is straightforward. We note that the viscous term in Eq. (2.11) is estimated explicitly with the Adams-Bashforth scheme, just for the purpose of calculating the forcing term $\hat{\boldsymbol{f}}_{ij}^{n+1/2}$. The reconstructed velocity $\boldsymbol{V}_{ij}^{n+1}$ at the forcing point is approximated using the velocities at the neighboring fluid points $\boldsymbol{u}^*$ and the control points on the rigid boundary. We note that $\boldsymbol{u}^*$ at the fluid points is obtained explicitly from Eq. (2.9) only for determining the forcing term at the forcing points.

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

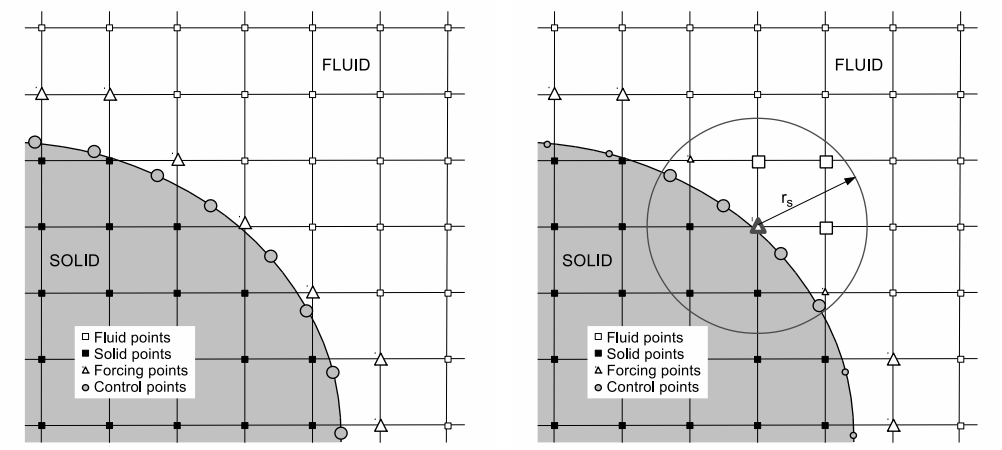


Figure 2: (a) Tagging of points in the computational domain around a rigid body. (b) Identification of points that are central to the moving least squares reconstruction.

More details of this direct-forcing approach can be found in [14] where a semi-implicit scheme is used in cylindrical coordinates. In the previous direct-forcing method [13], the reconstructed velocity $\boldsymbol{V}_{ij}^{n+1}$ at the forcing point is determined by interpolation along a specific direction such as $x-$,$y-$,$z-$directions or local normal directions. In the present study, we use an improved moving least squares reconstruction approach to calculate $\boldsymbol{V}_{ij}^{n+1}$ at the forcing point so that more points from the rigid interfaces and from the fluid can enter the reconstruction procedure. More details of the fundamentals of the improved MLS method can be found in [28]. To facilitate the reconstruction process, for each forcing point we define a support domain around it and identify the Eulerian fluid nodes and Lagrangian control points inside the support domain as shown in Fig. 2(b). In this figure, the support domain is a circle of radius r_s centered at the forcing point. Using the MLS method, a scalar quantity u at a forcing point $\boldsymbol{x}$ is calculated in its support domain as:

$$u(\boldsymbol{x})=\sum_{j=1}^{m}p_j(\boldsymbol{x})a_j(\boldsymbol{x})=\boldsymbol{p}^T(\boldsymbol{x})\boldsymbol{a}(\boldsymbol{x}), \tag{2.12}$$

where $\boldsymbol{p}(\boldsymbol{x})=[p_1(\boldsymbol{x}),\cdots,p_m(\boldsymbol{x})]^T$ is the weighted orthogonal basis function vector of length m and $\boldsymbol{a}(\boldsymbol{x})$ is its coefficient vector. We approximate $u(\boldsymbol{x})$ by determining the vector $\boldsymbol{a}(\boldsymbol{x})$ that minimizes the following weighted residual:

$$R=\sum_{i=1}^{n}W(\boldsymbol{x}-\boldsymbol{x}_i)\left[\boldsymbol{p}^T(\boldsymbol{x}_i)\boldsymbol{a}(\boldsymbol{x})-u(\boldsymbol{x}_i)\right]^2, \tag{2.13}$$

where $\boldsymbol{x}_i$ is the position of the Eulerian grid point or Lagrangian control point i in the interpolation stencil of n points, and $W(\boldsymbol{x}-\boldsymbol{x}_i)$ is a weight function which depends on

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

the distance between $\boldsymbol{x}$ and $\boldsymbol{x}_i$. In the present study, we use a cubic spline for the weight function as following:

$$W(\boldsymbol{x}-\boldsymbol{x}_i)=\begin{cases}\frac{2}{3}-4r^2+4r^3, & r\leq 0.5,\\ \frac{4}{3}-4r+4r^2-\frac{4}{3}r^3, & 0.5<r\leq 1.0,\\ 0, & \text{otherwise},\end{cases} \tag{2.14}$$

where $r=||\boldsymbol{x}-\boldsymbol{x}_i||/r_s$. Minimizing the residual functional R with respect to $\boldsymbol{a}(\boldsymbol{x})$, we obtain:

$$\boldsymbol{A}(\boldsymbol{x})\boldsymbol{a}(\boldsymbol{x})=\boldsymbol{B}(\boldsymbol{x})\boldsymbol{u}_s, \tag{2.15}$$

where

$$\begin{aligned}\boldsymbol{A}(\boldsymbol{x})&=\sum_{i=1}^{n}W(\boldsymbol{x}-\boldsymbol{x}_i)\boldsymbol{p}(\boldsymbol{x_i})\boldsymbol{p}^T(\boldsymbol{x_i}),\\ \boldsymbol{B}(\boldsymbol{x})&=[W(\boldsymbol{x}-\boldsymbol{x}_1)\boldsymbol{p}(\boldsymbol{x_1}) \quad \dots \quad W(\boldsymbol{x}-\boldsymbol{x}_n)\boldsymbol{p}(\boldsymbol{x_n})],\end{aligned}$$

and $\boldsymbol{u}_s=[u_1 \quad u_2 \quad \cdots \quad u_n]^T$ is a nodal vector of n points in the support domain. Combining Eqs. (2.12) and (2.15) we obtain the approximation of $u(\boldsymbol{x})$ as follows:

$$u(\boldsymbol{x})=\sum_{i=1}^{n}\Phi_i(\boldsymbol{x})u_i=\boldsymbol{\Phi}^T(\boldsymbol{x})\boldsymbol{u}_s, \tag{2.16}$$

where

$$\boldsymbol{\Phi}^T(\boldsymbol{x})=\boldsymbol{p}^T(\boldsymbol{x})\boldsymbol{A}(\boldsymbol{x})^{-1}\boldsymbol{B}(\boldsymbol{x}) \tag{2.17}$$

is the shape function vector for the forcing point $\boldsymbol{x}$. As shown in [28], the weighted orthogonal basis functions $\boldsymbol{p}=[p_j]$ can be constructed from the monomial basis functions $[1 \; x \; y \; z]$ for linear subspace and $[1 \; x \; y \; z \; x^2 \; y^2 \; z^2 \; xy \; xz \; yz]$ for quadratic subspace. For example, from the linear monomial basis functions $[q_1,q_2,q_3,q_4]=[1,x,y,z]$ the weighted orthogonal basis functions $\boldsymbol{p}=[p_j]$ can be obtained as

$$p_j=q_j-\sum_{k=1}^{j-1}\frac{(q_j,p_k)}{(p_k,p_k)}p_k, \qquad j=1,2,\cdots,m, \tag{2.18}$$

where (q_j,p_k) is an inner product defined as

$$(q_j,p_k)=\sum_{i=1}^{n}W(\boldsymbol{x}-\boldsymbol{x}_i)q_j(\boldsymbol{x_i})p_k(\boldsymbol{x_i}). \tag{2.19}$$

We note that the inner product of the weighted orthogonal basis functions (p_j,p_k) is only nonzero if $j=k$. Therefore the matrix $\boldsymbol{A}$ in (2.15) is a diagonal matrix with the component $A_{jj}=(p_j,p_j)$, $j=1,\cdots,m$ [28]. So in the improved MLS method [28], the shape function vector $\boldsymbol{\Phi}(\boldsymbol{x})$ of Eq. (2.17) can be obtained directly by simply inverting the diagonal matrix

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

$\boldsymbol{A}$. The choice of the support domain radius r_s is important to calculate $\boldsymbol{\Phi}(\boldsymbol{x})$ as it determines the number of Eulerian and Lagrangian grid points in the support domain. The suitable support domain must be sufficient large to guarantee the accuracy and efficiency of the MLS approximation. If the support domain is too large, the shape function vector $\boldsymbol{\Phi}(\boldsymbol{x})$ may be too smooth to approximate the local fluid properties and the computational cost will increase. If r_s is too small, the support domain will not contain enough points for interpolation or the smoothness of $\boldsymbol{\Phi}(\boldsymbol{x})$ will reduce. In our numerical experiments, we use the radius of the support domain r_s of about 1.5-2.5 grid sizes.

Once we have the shape functions at the forcing point $\boldsymbol{x}$, we can reconstruct its intermediate velocity based on the fluid nodes and boundary nodes in its support domain. We then compute the forcing term from Eq. (2.11). Using this forcing term, we can now solve Eq. (2.5) for the intermediate velocity field $\boldsymbol{u}^*$ and update the velocity and pressure fields for the next timestep. The overall algorithm will be summarized in the next section after the deformable boundaries are taken into account. It should be noted that boundary conditions for the pressure near the solid immersed boundaries are not imposed directly in the present direct-forcing approach but the whole pressure field is solved to satisfy the divergence-free condition (2.2). As clarified in [13,15], pressure boundary conditions are implicit into the right-hand side of the Poisson's equation (2.6). Solving the Poisson's equation (2.6) may result in spurious pressure values at the pressure solid points and pseudo-fluid points which do not influence the rest of the computation in case of the stationary immersed boundary. For moving rigid bodies, forcing points can become fluid points in the next timestep and field-extension procedure proposed in [15–17] can be employed to eliminate the unphysical velocity and pressure derivatives at the forcing points at the end of the current timestep.

2.3 Treatment of the deformable immersed boundaries

To handle the coupling between the fluid and the deformable boundary motion we employ the IB method [2] which represents the deformable surface by NL Lagrangian control points $\mathbf{X}_l$, $l=1,\cdots,$NL forming a triangular mesh. In the IB method, the elastic force per unit area $\boldsymbol{F}_l$ is determined at the Lagrangian points $\mathbf{X}_l$ and distributed to the Eulerian fluid points $\boldsymbol{x}$ using certain forms of the delta function,

$$\boldsymbol{f}(\boldsymbol{x},t)=\sum_{l=1}^{\mathrm{NL}}\boldsymbol{F}_l(\mathbf{X}_l,t)D_h(\boldsymbol{x}-\mathbf{X}_l(t))\triangle S_l, \tag{2.20}$$

where $\triangle S_l$ is the local area at $\mathbf{X}_l$, $D_h(\boldsymbol{x})$ is a 3-D discrete delta function [2]. The discrete delta functions constructed on uniform grid work well when the deformable boundaries are not very close to the immersed rigid boundaries. However, when the deformable boundaries are within 2 grid points from the rigid boundaries, the discrete delta functions may spread the force into the solid region and this force does not contribute to the motion of the fluid in the direct-forcing method. To avoid this, we employ the MLS shape

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

functions constructed in a similar manner as those mentioned in the previous section for the distribution of Lagrangian forces and the interpolation of velocity. To facilitate the construction of the MLS shape functions, for each Lagrangian control point $\mathbf{X}_l$ of the deformable boundary, we define a support domain around it and identify the background Eulerian grid points inside the support domain excluding the points in the solid regions. Once we have identified *ne* qualified Eulerian points in the support domain of $\mathbf{X}_l$, we can compute the MLS shape functions using Eq. (2.17). By definition, the shape functions satisfy the properties

$$\sum_{k=1}^{ne} \Phi_k^l(\mathbf{X}_l) = 1, \tag{2.21}$$

and

$$\sum_{k=1}^{ne} \Phi_k^l(\mathbf{X}_l) \boldsymbol{x}_k = \mathbf{X}_l, \tag{2.22}$$

for all Lagrangian points $\mathbf{X}_l$ [34]. With the introduction of shape functions, the velocity at the Lagrangian point $\mathbf{X}_l$ can be interpolated as

$$\boldsymbol{U}_l(\mathbf{X}_l) = \sum_{k=1}^{ne} \Phi_k^l(\mathbf{X}_l) \boldsymbol{u}_k, \tag{2.23}$$

and the force at the Lagrangian point $\mathbf{X}_l$ can be distributed to the surrounding fluid points as

$$\boldsymbol{f}_k = \frac{1}{h^3} \sum_{l=1}^{NL} \Phi_k^l(\mathbf{X}_l) \boldsymbol{F}_l \triangle S_l, \tag{2.24}$$

where h is Cartesian grid size.

It can be shown that the power generated by the elastic force is the same whether it is evaluated in the Eulerian or Lagrangian meshes. From the power generated by the Eulerian elastic force, we find

$$\begin{aligned} \sum_{k=1}^{tne} \boldsymbol{f}_k \cdot \boldsymbol{u}_k h^3 = \sum_{k=1}^{tne} \sum_{l=1}^{NL} \Phi_k^l \boldsymbol{F}_l \triangle S_l \cdot \boldsymbol{u}_k &= \sum_{l=1}^{NL} \boldsymbol{F}_l \cdot \left(\sum_{k=1}^{tne} \Phi_k^l \boldsymbol{u}_k \right) \triangle S_l \\ &= \sum_{l=1}^{NL} \boldsymbol{F}_l \cdot \boldsymbol{U}_l \triangle S_l, \end{aligned} \tag{2.25}$$

which is the corresponding Lagrangian expression for the elastic power. Here, *tne* is the total Eulerian grid points in the fluid zone and no specific properties of Φ_k^l are required for the above power identity. We first use Eq. (2.24) to replace $\boldsymbol{f}_k$ and then we use Eq. (2.23) to eliminate $\boldsymbol{u}_k$ with the fact that $\Phi_k^l(\mathbf{X}_l)$ is only nonzero at *ne* grid points inside the support domain of $\mathbf{X}_l$. Using Eq. (2.21), we can show that the total force is conserved when transfer from Lagrangian to Eulerian meshes:

$$\sum_{k=1}^{tne} \boldsymbol{f}_k h^3 = \sum_{k=1}^{tne} \sum_{l=1}^{NL} \Phi_k^l \boldsymbol{F}_l \triangle S_l = \sum_{l=1}^{NL} \left(\sum_{k=1}^{tne} \Phi_k^l \right) \boldsymbol{F}_l \triangle S_l = \sum_{l=1}^{NL} \boldsymbol{F}_l \triangle S_l. \tag{2.26}$$

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

In the same way but relying on Eq. (2.22), we can also show the torque identity between the Eulerian and Lagrangian meshes

$$\sum_{k=1}^{tne} \boldsymbol{x}_k \times \boldsymbol{f}_k h^3 = \sum_{k=1}^{tne} \boldsymbol{x}_k \times \sum_{l=1}^{NL} \Phi_k^l \boldsymbol{F}_l \triangle S_l = \sum_{l=1}^{NL} \left(\sum_{k=1}^{tne} \boldsymbol{x}_k \Phi_k^l \right) \times \boldsymbol{F}_l \triangle S_l$$
$$= \sum_{l=1}^{NL} \mathbf{X}_l \times \boldsymbol{F}_l \triangle S_l. \tag{2.27}$$

In summary, the interpolation defined by (2.23) and distribution operation defined by (2.24) have established the equivalence between the Eulerian and Lagrangian formulations for the force, torque and power. And therefore these two equations guarantee the conservation of the momentum, angular momentum and energy during the transfer.

Using a trapezoidal rule, we compute the new location of the deformable boundary, $\boldsymbol{X}_l$, as

$$\boldsymbol{X}_l^{n+1} = \boldsymbol{X}_l^n + \triangle t \left(\boldsymbol{u}^{n+1/2} \left(\boldsymbol{X}_l^{n+1/2} \right) \right), \tag{2.28}$$

where $\boldsymbol{u}^{n+1/2}$ and $\boldsymbol{X}_l^{n+1/2}$ are time average velocity and position of the control point, respectively. Eq. (2.28) is a non-linear system of equations which can be rewritten as

$$\boldsymbol{g}\left(\boldsymbol{X}_l^{n+1} \right) = \mathbf{0}, \tag{2.29}$$

where

$$\boldsymbol{g}(\boldsymbol{X}_l) = \boldsymbol{X}_l - \boldsymbol{X}_l^n - \triangle t \left(\boldsymbol{u}^{n+1/2} \left(\frac{\boldsymbol{X}_l^n + \boldsymbol{X}_l}{2} \right) \right), \tag{2.30}$$

and can be solved by a Newton-Krylov method [35, 36].

2.4 Implementation

The current moving least-square immersed boundary method can be described as following:

1. represent solid and deformable boundaries using different sets of control points that define a triangular unstructured meshes;
2. compute the elastic forces on the deformable boundaries;
3. calculate the forcing terms at the forcing points in the neighborhood of the solid objects;
4. compute the fluid velocity and pressure using projection method;
5. determine the new position of the deformable boundaries.

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

In our implementation, the elastic force $\boldsymbol{F}$ generated on the deformable surface during its deformation is calculated using a thin-shell model with subdivision surface [37, 38]. For solid bodies, the forcing points are identified in the immediate vicinity of the solid boundaries. The MLS shape functions are built for each of the forcing points. We note that the forcing points and their corresponding shape functions are only constructed once for stationary solid boundaries. In contrast, as the control points of deformable boundaries move with the fluid, their corresponding MLS shape functions have to be reconstructed with the updated locations of the deformable boundaries. Since the computational cost for constructing MLS shape functions is more expensive than the cost of using the traditional discrete delta function, we only use MLS shape functions for distributing forces and interpolating velocities at a few control points closed to the solid boundary. For the rest of the control points on the deformable boundaries, we still use the traditional discrete delta function for the distribution and interpolation procedures to save the computational cost. And it can be proven that the use of both MLS shape functions and discrete delta function to transfer force and velocity between Eulerian and Lagrangian meshes can still guarantee the conservation of the momentum, angular momentum and energy during the transfer.

Once the elastic forces are computed over the surfaces of the deformable boundaries and distributed to the Eulerian grid points in the fluid domain, the forcing terms are enforced at the vicinity of the solid boundaries and the N-S equations can be solved for the fluid pressure and velocity fields. We then update the deformable boundaries using Eq. (2.28). In summary, given $\boldsymbol{X}^n$, $\boldsymbol{u}^n$, $p^{n-1/2}$, the forcing points and their corresponding MLS shape functions, we can compute the new $\boldsymbol{X}^{n+1}$, $\boldsymbol{u}^{n+1}$, $p^{n+1/2}$ and enforce the noslip condition at the rigid boundaries as following:

Step 1: Set $k=0$ and make a guess for $\boldsymbol{X}^{n+1}$, i.e. $\boldsymbol{X}^{(0)}=2\boldsymbol{X}^n-\boldsymbol{X}^{n-1}$.

Step 2: For deformable boundaries:

- Compute $\boldsymbol{X}^{n+1/2}=(\boldsymbol{X}^n+\boldsymbol{X}^{(k)})/2$ and evaluate the elastic forces $\boldsymbol{F}^{n+1/2}$ on the surfaces of the deformable boundaries $\boldsymbol{X}^{n+1/2}$ [38, 39].
- Using the MLS shape functions or discrete delta functions to obtain $\boldsymbol{f}^{n+1/2}$.

Step 3: For solid boundaries:

- Perform an explicit predictor step to compute the intermediate velocity $\boldsymbol{u}^*$ at the Eulerian fluid points.
- Reconstruct the velocity at the forcing points using MLS interpolation and calculate the corresponding forcing terms.

Step 4:

- Solve for $\boldsymbol{u}^{n+1}$ and $p^{n+1/2}$ as shown in Section 2.1.
- Calculate $\boldsymbol{u}^{n+1/2}(\boldsymbol{X}^{n+1/2})$ of the deformable boundaries.

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

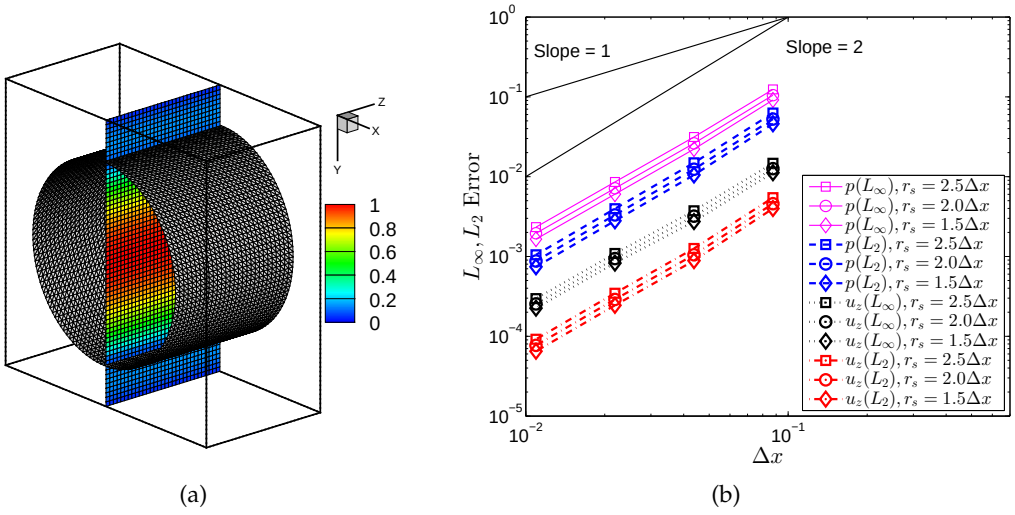
Step 5:

- Compute $g\left(\boldsymbol{X}^{(k)}\right)=\boldsymbol{X}^{(k)}-\boldsymbol{X}^{n}-\frac{1}{2}\triangle t\boldsymbol{u}^{n+1/2}(\boldsymbol{X}^{n+1/2})$.
- Set $\boldsymbol{X}^{n+1}=\boldsymbol{X}^{(k)}$ if $\|g\left(\boldsymbol{X}^{(k)}\right)\|<\epsilon$. Otherwise, update $\boldsymbol{X}^{(k+1)}$ by solving Eq. (2.29) [36] and repeat the iteration from step 2.

3 Results

3.1 Poiseuille flow in a cylindrical tube

To show the accuracy of the IB method with MLS interpolation for rigid boundary, we apply it to solve for the Poiseuille flow in a cylindrical tube of radius R. The exact solutions are $u_x=u_y=0, u_z(r)=U_{\max}\left[1-(\frac{r}{R})^2\right]$, where $U_{\max}=-\frac{R^2}{4\mu}\frac{dp}{dz}$ is the maximum (centerline) velocity. The cylindrical tube is immersed in a computational domain of size $2.8R\times 2.8R\times 1.4R$ as show in Fig. 3(a). The present MLS reconstruction approach is used to impose the noslip boundary condition on the cylinder surface. We perform simulations for 4 different meshes with $32^2\times 16, 64^2\times 32, 128^2\times 64, 256^2\times 128$ grid points. For all cases, we set $R=1$, $\mu=1$ and $U_{\max}=1$. The Dirichlet and Neumann boundary conditions for velocity are set at the inlet and outlet faces respectively while the noslip boundary conditions for velocity are set at the other faces of the computational domain. The initial conditions are $\boldsymbol{u}=\boldsymbol{0}$ and $p=0$. Once the solutions reach the steady state, we measure the



(a) (b)

Figure 3: Poiseuille flow in a circular tube. (a) Computational set up and the obtained velocity field. (b) L_2- and L_∞-norms of the velocity and pressure errors at three radii of support domain.

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

error of the velocity and pressure at different spatial resolution in L_2- and L_∞-norms as

$$L_2 = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\left(\psi_k - \psi_k^e\right)^2}, \qquad L_\infty = \max_{k=1,\cdots,N}|\psi_k - \psi_k^e|, \tag{3.1}$$

where ψ_k^e and ψ_k are the analytical and numerical solutions respectively at point k such as velocity and pressure; N is the number of grid point in the fluid region. For velocity points, N includes the forcing points and the fluid points. For pressure points, N does not contain the pseudo-fluid points as we do not reconstruct the pressure at these points. In Fig. 3(b), it can be seen that second-order accuracy can be achieved for both velocity and pressure in problems with only rigid boundaries using the MLS reconstruction with linear basis functions. If we use the MLS with second order monomials as the basis functions, we can reach the exact solution up to the machine error since the exact velocity is a quadratic function in space. In Fig. 3(b), we calculate the errors at three different radii of support domain $r_s = 1.5$, 2 and 2.5 grid sizes. The errors reduce slightly with reducing support domain radius because the shape function constructed on a smaller support domain can represent local properties of the fluid better. But reducing r_s below $1.5\Delta x$ will result in an inaccurate function approximation since there are not enough grid points in the support domain.

3.2 Deformation of single capsule in shear flow

Next, we study the performance of the MLS interpolation and distribution operators in the treatment of the deformable boundaries. We investigate the deformation of a spherical elastic capsules enclosed by neo-Hookean thin shell in a simple shear flow with a shear rate $\dot{\gamma}$. The capsule deformation is described by $D_{xy} = (L-B)/(L+B)$, where B and L are the minimum and maximum distances from the origin, respectively [40] as shown in Fig. 4. The capsule deformation depends on several dimensionless parameters

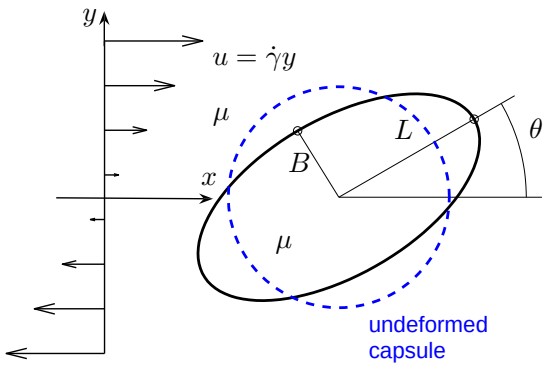
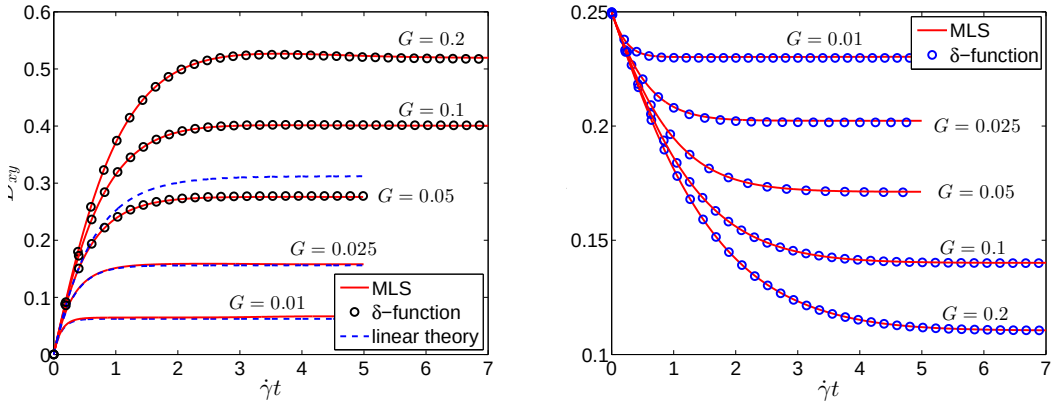


Figure 4: Sketch of an elastic capsule in shear flow. The dashed line is the initial undeformed capsule.

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press



(a) (b)

Figure 5: Evolution history of (a) D_{xy} and (b) θ of a spherical capsule.

such as shear rate $G = \mu\dot{\gamma}a/E_s$, bending modulus $\hat{\kappa}_B = \kappa_B/(a^2E_s)$ and Reynolds number $Re = \frac{\rho\dot{\gamma}a^2}{\mu}$ where a is the capsule radius, E_s and κ_B are surface elastic modulus and bending modulus respectively. In this example, $Re = 0.01$, $\hat{\kappa}_B = 0$ are chosen so that the capsule deformation is able to compared with that from small deformation theory [41] and the IB method with delta function [38]. Computational domain is a cube domain of size $10a$ with prescribed velocity boundary condition at the boundaries $y = \pm 5a$ and periodic boundary condition at other boundaries. The fluid domain is discretized with a $96\times96\times96$ uniform grid and the capsule membrane is discretized using a triangular mesh of 20480 elements and 10242 nodes. The average ratio between the Lagrangian and Eulerian mesh sizes is about 0.35 which guarantees the condition for preventing volume loss in IB method. Fig. 5 plots the time history of the deformation parameter, D_{xy}, and inclination angle, θ, for different G. Good agreement with the linear theory [41] can be seen in Fig. 5(a) for small G ($G<0.025$). As G increases the deformation becomes large and the linear theory is no longer valid. Hence, we compare the present results with those obtained using the immersed boundary method with delta function [38]. We note that the only different between the present numerical method and that in [38] is the use of MLS and delta function for velocity interpolation and force distribution. We can see excellent agreement in both deformation and inclination between the MLS and delta function methods. In this example, the MLS method does not show any advantages over the traditional discrete delta function method. In fact the computational cost of using the MLS method for velocity interpolation and force distribution is more expensive than that of the traditional discrete delta function method. The total computational time is about 9% higher if we use the MLS shape functions constructed with a support domain of radius $r_s = 2\Delta x$ instead of using the 4-point discrete delta function on an Intel Xeon 2.9GHz. We only use this example to show that the shape functions constructed by the MLS method

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

can replace the discrete delta function in Peskin's IB method and the MLS approach can be used to design an immersed boundary method on unstructured meshes.

3.3 Red blood cell in microfluidics devices

In this example, we study the motion of a single red-blood cell (RBC) in deterministic lateral displacement (DLD) devices used to separate particles or cells according to sizes [42]. The DLD devices comprise of columns of micron-sized pillars whose arrangement is illustrated in Fig. 6. Particles and cells are flowed through this device from left to right and they are sorted at the outlet of the device based on their sizes and deformation [43, 44]. If the effective size of the particles is smaller than a certain critical diameter, they will move through the gaps between micropillars in a zigzag manner without bumping into the downstream pillars. In contrast, if the effective size of the particles is larger than a certain critical size, they will exhibit a displaced trajectory [43] and move diagonally along the device. The key parameters governing the flow characteristics through the DLD devices are the micropillar radius R, the horizontal and vertical distances between two micropillars λ and the row shift d. The behavior and trajectories of the RBC in the DLD devices depend on the capillary number $Ca=\mu u_{\max}/E_s$, reduced bending modulus $\hat{\kappa}_B$ and $Re=\rho u_{\max} a/\mu$ where a is the RBC radius and $u_{\max}$ is the maximum velocity in the DLD device without the RBC presence. Here, we describe the biconcave disk shape of the RBC at rest using the mapping in [45] and employ the Skalak's strain energy function [46] for RBC deformation.

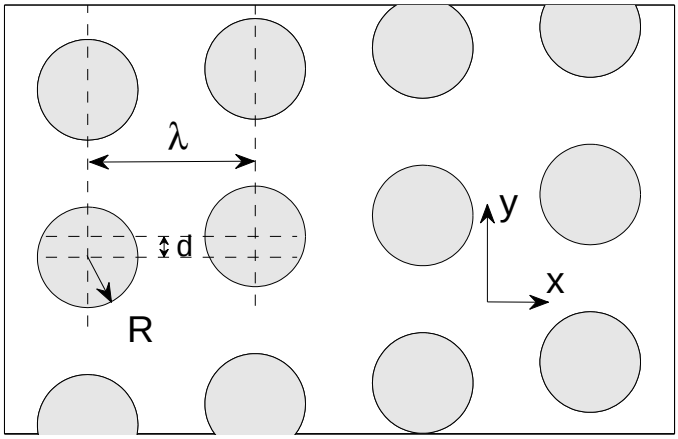


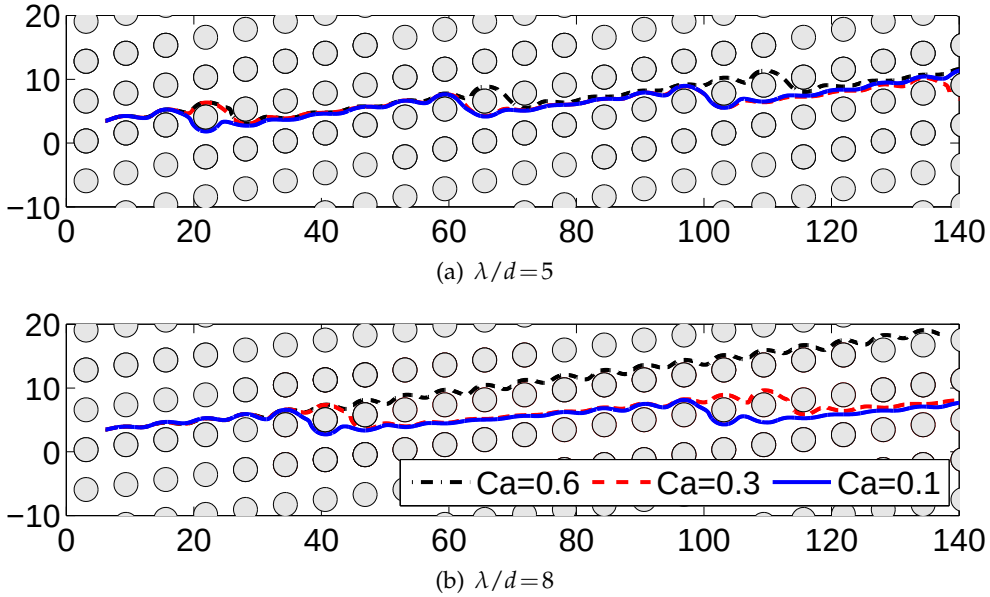
Figure 6: Arrangement of the micropillars in the DLD devices. The geometry is defined by the micropillar radius R, the horizontal and vertical distances between two micropillars λ and the row shift d.

In our simulations, we have fixed $R/a=1.875$, $\lambda/a=6.25$, channel height $L_z=3.125a$, channel width $L_y=12.5a$. These geometric parameters of the DLD devices are similar to those studied in [47]. The length of the channel is chosen depending on the row

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

shift d. We perform simulations for two different DLD devices with $d/\lambda=0.125$ and 0.2 which mean that the pillar arrangement is repeated after every $\lambda/d=8$ and 5 columns, respectively. The channel lengths for these two DLD devices are $L_x=8\lambda$ and 5λ, respectively. The computational domains are discretized with uniform Cartesian grid of gridsize $h=L_z/40$. The surfaces of the micropillars are discretized with a rectangular mesh of similar meshsize h and the RBC membrane is discretized with a triangular mesh of 20480 elements and 10242 nodes. Periodic boundary condition is applied in $x-$ and $y-$directions and noslip condition is prescribed at the bottom and top walls. The RBC which exits the computational domain in the $x-$ or $y-$directions re-enters the domain from the opposite sides. For the two different DLD devices, two corresponding pressure drops ΔP are applied to generate the same maximum velocity $u_{\max}$ inside the channel. In this example, we use the traditional discrete delta function for the IB points whose interpolation stencils lie entirely within the fluid and MLS shape functions for the IB points in the vicinity of the rigid boundaries to save the computational cost.

We performed simulations for three different capillary numbers Ca=0.1,0.3 and 0.6 at the fixed Re=0.01 and $\hat{\kappa}_B$=0.01. Fig. 7 shows two types of trajectories, namely, displaced and zigzag trajectories as observed experimentally. In Fig. 7, the center of mass of the red-blood cell is plotted against time as it moves through the DLD devices. Fig. 7(a) shows the zigzag trajectories of the RBC with different capillary numbers in the DLD device with the row shift $\lambda/d=5$. In this configuration the effective size of the RBC is relative small compared to the row shift d and it can pass through 5 columns of micropillars with-

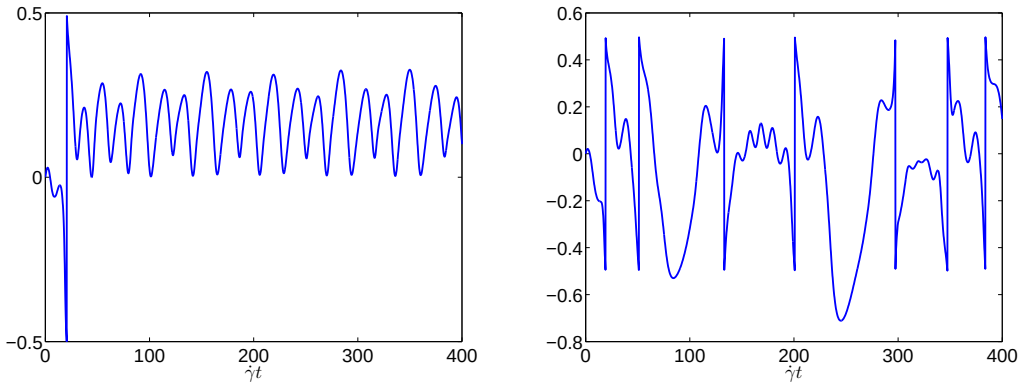


(a) $\lambda/d=5$

(b) $\lambda/d=8$

Figure 7: Trajectories of the RBC at different capillary numbers $Ca=0.6,0.3,0.1$ in two DLD devices with different row shifts.

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press



(a) (b)

Figure 8: Inclination angle of the RBC in DLD device with the row shift $\lambda/d=8$. (a) $Ca=0.6$: tank-treading and (b) $Ca=0.1$: tumbling.

out bumping into any of the micropillars. Fig. 7(b) shows both the zigzag and displaced trajectories of the RBC with different capillary numbers in the DLD device with the row shift λ/d=8. In this configuration, the row shift d=0.78a is comparable to the thickness of the RBC at rest ($\approx$0.66a) which is about the effective size of the RBC under deformation. The RBC exhibits the displaced trajectory with $Ca=0.6$ and the zigzag trajectories with Ca=0.3 and 0.1. At Ca=0.6, while undergoing the displaced trajectory along the channel the RBC exhibits a tank-treading motion [48] and its inclination angle oscillates periodically as shown in Fig. 8(a). This type of motion is called swinging motion as observed experimentally in [49] for simple shear flow. At $Ca=0.3$ and 0.1, the RBC exhibits the zigzag trajectories while undergoing continuous rotation. The time variation of θ with $Ca=0.1$ in Fig. 8(b) shows the tumbling motion and rotation around the micropillars. The two local minima with $\theta<-\pi/2$ in Fig. 8(b) correspond to the two rotations around the micropillars in the zigzag trajectories as shown in Fig. 7(b) for $Ca=0.1$. Therefore we can see that the dynamics of RBCs under shear flow (swinging or tumbling) also affect the trajectories (displaced or zigzag) of the RBCs in the DLD devices in addition to their effective sizes.

Fig. 9 shows different positions of the RBC along the displaced trajectory in the DLD device to help to visualize better the swing motion of the RBC. The black dot which represents the same surface point of the RBC membrane rotates around the inner fluid periodically while the RBC undergoes swinging motion. The swinging motion with oscillated inclination angle may enlarge the effect size of the RBC which leads to the displaced trajectory of RBC in the DLD device.

Fig. 10 visualizes the tumbling motion and the rotation around a micropillar during the zigzag trajectory of the RBC in the DLD device. In this figure, the RBC has performed a full rotation from dimensionless time $t=0$ to $t=65$. And the black dots represent the same surface point which remains at the same dimple location on the membrane during

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

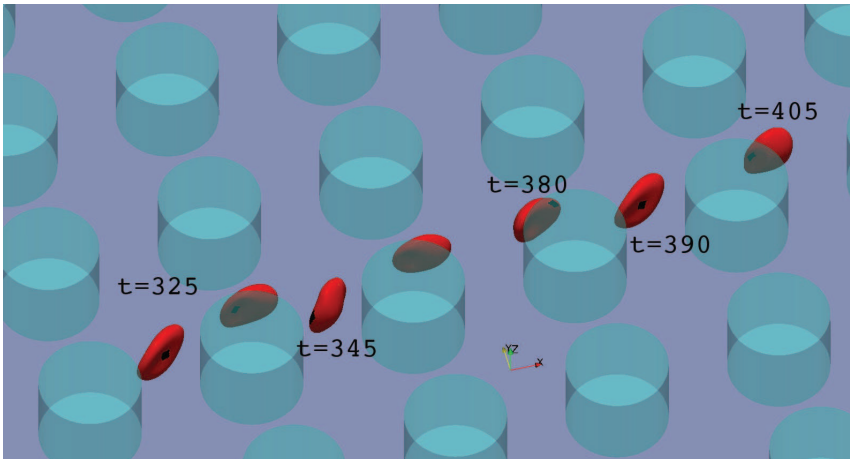


Figure 9: Shapes of the RBC at different times during a tank-treading motion at $Ca=0.6$. The black dot on the surface of the RBC corresponds to the same point of the RBC membrane at different time steps.

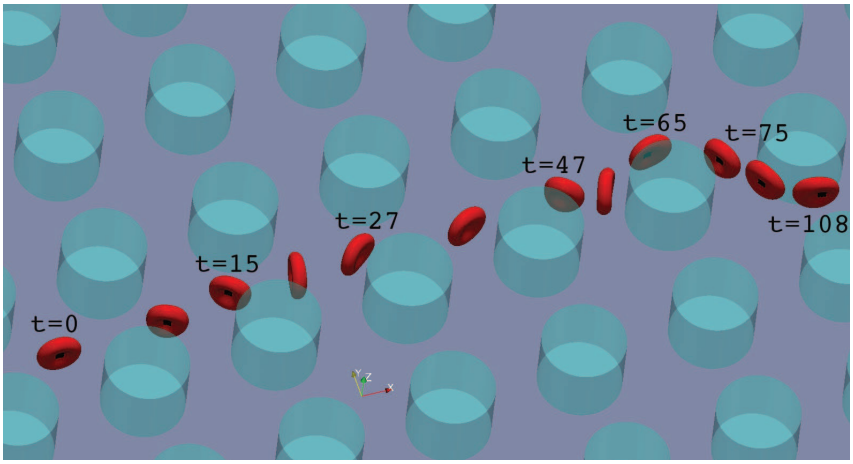


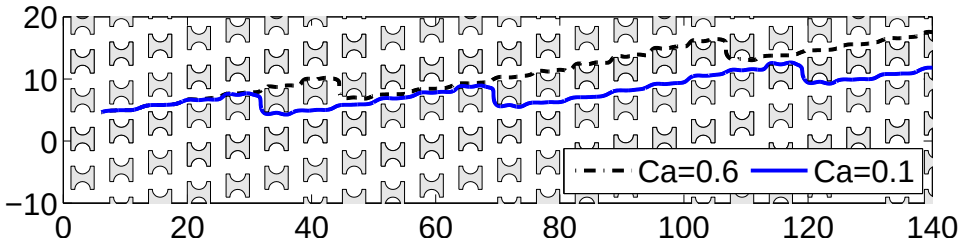
Figure 10: Shapes of the RBC at different times during a tumbling motion at $Ca=0.1$. The black dot on the surface of the RBC corresponds to the same point of the RBC membrane at different time steps.

the tumbling motion. The tumbling motion of the RBC significantly perturbs the local streamlines and allow the RBC to enter the gap between the two micropillars on the same row easily at a suitable inclination angle. As a result, RBC that undergoes tumbling motion at low capillary numbers tends to exhibit the zigzag trajectory. The RBC rotates around a micropillar between $t=75$ and $t=108$ which corresponds to the time interval around the first local minimum with $\theta<-\pi/2$ in Fig. 8(b). We note that simulations have been performed for different initial positions of the RBC in the DLD devices and we see the same trajectories and dynamics of the RBC. Thus, RBCs with the same capillary

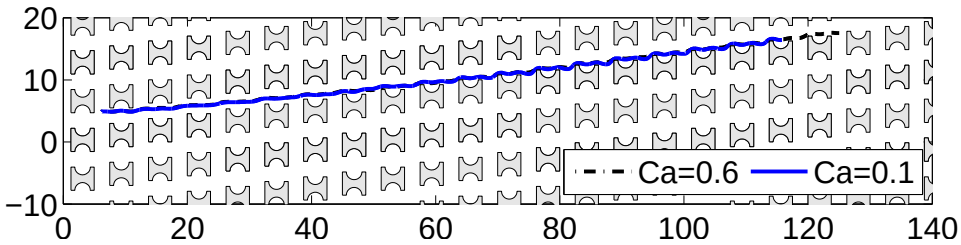
https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

number will exhibit the same type of trajectory and behaviour regardless of their initial positions.

Next, we also simulate the motion of RBC in DLD devices with I-shaped micropillars as studied in [47]. This configuration demonstrates that our proposed IB method with MLS interpolation works well for complex geometry and interaction between the deformable and rigid boundaries. In DLD devices with I-shaped micropillars, we observe similar trajectories of the RBC as those in DLD devices with circular pillars. Fig. 11(a) shows the zigzag trajectories of the RBC with $Ca=0.1$ and 0.6 in the DLD device with the row shift $\lambda/d=5$. And Fig. 11(b) shows the displaced trajectories of the RBC at two different capillary numbers in the DLD device with the row shift $\lambda/d=8$. In contrast to the DLD devices with circular pillars, the RBC in the I-shaped device with the row shift $\lambda/d=8$ only exhibit displaced trajectories. This result shows that the I-shaped DLD devices can displace the RBC better than the circular devices. And this result is consistent with the experimental observation in [47] where the I-shaped micropillars are used to enhance the separation efficiency.



(a) $\lambda/d=5$



(b) $\lambda/d=8$

Figure 11: Trajectories of the RBC at different dimensionless shear rates $Ca=0.6$ and 0.1 in two DLD devices with different row shifts.

4 Conclusions

We have presented a moving-least-square Immersed Boundary method for viscous flow involving deformable and rigid boundaries. For rigid boundaries, the MLS method was implemented with the direct-forcing method to allow more rigid and fluid points enter-

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

ing the reconstruction process given the strong viscous effect of the fluid. The present method involving rigid boundaries was validated by studying the pressure-driven flow in a tube. Excellent agreement between exact and numerical solutions have been demonstrated in this example. For deformable boundaries, the same MLS method was implemented to construct the interpolation and distribution operators to avoid unphysical errors when the deformable boundaries are close to the rigid boundaries. The present interpolation and distribution operators constructed by the MLS method are compared to those using the discrete delta functions by investigating the deformation of the spherical elastic capsule in shear flow. Our results demonstrate good agreement with those obtained by theory and other numerical studies. Finally, simulations for the RBC dynamics in a cell sorter over a range of capillary numbers and different device configurations have been performed. Different types of RBC trajectories such as zig-zag and displaced trajectories have been observed depending on the capillary number and row-shift of the device. We have also simulated the motion of RBC in DLD devices with I-shaped micropillars to show the robustness of the present approach in handing fluid-structure interaction in complicated domain. The trajectories and behaviours of the RBC in the DLD devices are in consistent with the experimental observation.

References

[1] C. S. Peskin, Numerical analysis of blood flow in the heart, J. Comput. Phys. 25 (1977) 220–252.
[2] C. S. Peskin, The immersed boundary method, Acta Numerica 11 (2) (2002) 479–517.
[3] L. J. Fauci, C. S. Peskin, A computational model of aquatic animal locomotion, J. Comput. Phys 77 (1988) 85–108.
[4] C. D. Eggleton, A. S. Popel, Large deformation of red blood cell ghosts in a simple shear flow, Phys. Fluids 10 (1998) 1834–1845.
[5] R. Dillon, L. J. Fauci, D. Graver, A microscale model of bacterial swimming, chemotaxis and substrate transport, J. Theor. Biol. 177 (1995) 325–340.
[6] N. T. Wang, A. L. Fogelson, Computational methods for continuum models of platelet aggregation, J. Comput. Phys 151 (1999) 649–675.
[7] M. C. Lai, C. S. Peskin, An immersed boundary method with formal second order accuracy and reduced numerical viscosity, J. Comput. Phys. 160 (2000) 707–719.
[8] A. L. E. Silva, A. Silveira-Neto, J. Damasceno, Numerical simulation of two-dimensional flows over a circular cylinder using the immersed boundary method, J. Comput. Phys. 189 (2003) 351–370.
[9] D. V. Le, B. C. Khoo, K. M. Lim, An implicit-forcing immersed boundary method for simulating viscous flows in irregular domains, Comput. Methods Appl. Mech. Engrg. 197 (2008) 2119–2130.
[10] S. W. Su, M. C. Lai, C. A. Lin, An immersed boundary technique for simulating complex flows with rigid boundary, Comput. Fluids 36 (2007) 313–324.
[11] J. Mohd-Yusof, Combined immersed boundary/B-splines methods for simulations of flows in complex geometry, Annual Research Briefts, Center for Turbulence Research (1997) 317–327.

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

[12] E. A. Fadlun, R. Verzicco, P. Orlandi, Combined immersed-boundary finite-difference methods for three-dimensional complex flows simulations, J. Comput. Phys. 161 (2000) 35–60.
[13] E. Balaras, Modeling complex boundaries using an external force field on fixed Cartesian grids in large-eddy simulations, Computers & Fluids 33 (2004) 375–404.
[14] E. Balaras, J. Yang, Nonboundary conforming methods for large-eddy simulations of biological flows, ASME J. Fluids Eng. 127 (2005) 851–857.
[15] J. Yang, E. Balaras, An embedded-boundary formulation for large-eddy simulation of turbulent flows interacting with moving boundaries, J. Comput. Phys. 215 (2006) 12–40.
[16] J. Yang, F. Stern, A simple and efficient direct forcing immersed boundary framework for fluid-structure interactions, J. Comput. Phys. 231 (2012) 5029–5061.
[17] C. Liu, C. Hu, An efficient immersed boundary treatment for complex moving object, J. Comput. Phys. 274 (2014) 654–680.
[18] M. Uhlmann, An immersed boundary method with direct forcing for the simulation of particulate flows, J. Comput. Phys. 209 (2005) 448–476.
[19] H. S. Udaykumar, R. Mittal, P. Rampunggoon, A. Khanna, A sharp interface Cartesian grid method for simulating flows with complex moving boundaries, J. Comput. Phys. 174 (2001) 345–380.
[20] T. Ye, R. Mittal, H. S. Udaykumar, W. Shyy, An accurate Cartesian grid method for viscous incompressible flows with complex immersed boundary, J. Comput. Phys. 156 (1999) 209–240.
[21] X. Hu, B. C. Khoo, N. Adams, F. Huang, A conservative interface method for compressible flows, J. Comput. Phys. 219 (2006) 553–578.
[22] Y. H. Tseng, J. H. Ferziger, A ghost-cell immersed boundary method for flow in complex geometry, J. Comput. Phys. 192 (2003) 593–623.
[23] L. Ge, F. Sotiropoulos, A numerical method for solving the 3D unsteady incompressible Navier-Stokes equations in curvilinear domains with complex immersed boundaries, J. Comput. Phys. 225 (2007) 1782–1809.
[24] I. Borazjani, L. Ge, F. Sotiropoulos, Curvilinear immersed boundary method for simulating fluid structure interaction with complex 3D rigid bodies, J. Comput. Phys. 227 (2008) 7587–7620.
[25] F. Roman, E. Napoli, B. Milici, V. Armenio, An improved immersed boundary method for curvilinear grids, Comput. Fluids 38 (2009) 1510–1527.
[26] R. Mittal, G. Iaccarino, Immersed boundary methods, Annu. Rev. Fluid Mech. 37 (2005) 239–261.
[27] F. Sotiropoulos, X. Yang, Immersed boundary methods for simulating fluid-structure-interaction, Progress in Aerospace Sciences 65 (2014) 121.
[28] K. Liew, Y. Cheng, S. Kitipornchai, Boundary element-free method (BEFM) for two-dimensional elastodynamic analysis using Laplace transform, Int. J. Numer. Meth. Engng 64 (2005) 1610–1627.
[29] M. Vanella, E. Balaras, A moving-least-squares reconstruction for embedded-boundary formulations, J. Comput. Phys. 228 (2009) 6617–6628.
[30] D. Li, A. Wei, K. Luo, J. Fan, An improved moving-least-squares reconstruction for immersed boundary method, Int. J. Numer. Meth. Engng 104 (2015) 789–804.
[31] D. L. Brown, R. Cortez, M. L. Minion, Accurate projection methods for the incompressible Navier-Stokes equations, J. Comput. Phys. 168 (2001) 464–499.
[32] J. Adams, P. Swarztrauber, R. Sweet, FISHPACK: Efficient FORTRAN subprograms for the solution of separable elliptic partial differential equations, Available on the web at

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press

http://www.scd.ucar.edu/css/software/fishpack/.
[33] U. Schumann, R. A. Sweet, A direct method for the solution of Poisson's equation with Neumann boundary conditions on a staggered grid of arbitrary size, J. Comput. Phys. 20 (1976) 171–182.
[34] G.-R. Liu, Y.-T. Gu, An introduction to meshfree methods and their programming, Springer, 2005.
[35] D. A. Knoll, D. E. Keyes, Jacobian-free Newton-Krylov methods: a survey of approaches and applications, J. Comput. Phys 193 (2004) 357–397.
[36] D. V. Le, J. White, J. Peraire, K. M. Lim, B. C. Khoo, An implicit immersed boundary method for three-dimensional fluid-membrane interactions, J. Comput. Phys. 228 (2009) 8427–8445.
[37] F. Cirak, M. Ortiz, P. Schroder, Subdivision surfaces: a new paradigm for thin-shell finite-element analysis, Int. J. Numer. Meth. Engng. 47 (2000) 2039–2072.
[38] D. V. Le, Subdivision elements for large deformation of liquid capsules enclosed by thin shells, Comput. Methods Appl. Mech. Engrg. 199 (2010) 2622–2632.
[39] D. V. Le, Z. Tan, Hydrodynamic interaction of elastic capsules in bounded shear flow, Commun. Comput. Phys. 16 (2014) 1031–1055.
[40] S. Ramanujan, C. Pozrikidis, Deformation of liquid capsules enclosed by elastic membrane in simple shear flow: large deformations and the effect of fluid viscosities, J. Fluid Mech. 361 (1998) 117–143.
[41] D. Barthès-Biesel, J. M. Rallison, The time-dependent deformation of a capsule freely suspended in a linear shear flow, J. Fluid Mech. 113 (1981) 251–267.
[42] L. R. Huang, E. C. Cox, R. H. Austin, J. C. Sturm, Continuous particle separation through deterministic lateral displacement, Science 304 (2004) 987.
[43] J. A. Davis, D. W. Inglis, K. J. Morton, D. A. Lawrence, L. R. Huang, S. Y. Chou, J. C. Sturm, R. H. Austin, Deterministic hydrodynamics: Taking blood apart, PNAS 103 (2006) 14779–14784.
[44] R. Quek, D. V. Le, K. H. Chiam, Separation of deformable particles in deterministic lateral displacement devices, Phys. Rev. E 83 (2011) 056301.
[45] E. Evans, Y. C. Fung, Improved measurements of the erythrocyte geometry, Microvasc. Res. 4 (1972) 335–347.
[46] R. Skalak, A. Tozeren, R. P. Zarda, S. Chien, Strain energy function of red blood cell membranes, Biophys. J. 13 (1973) 245–264.
[47] K. K. Zeming, S. Ranjan, Y. Zhang, Rotational separation of non-spherical bioparticles using I-shaped pillar arrays in a microfluidic device, Nature Communications 4 (2013) 1625.
[48] J. M. Skotheim, T. W. Secomb, Red blood cells and other nonspherical capsules in shear flow: Oscillatory dynamics and the tank-treading-to-tumbling transition, Phys. Rev. Lett. 98 (2007) 078301.
[49] M. Abkarian, M. Faivre, A. Viallat, Swinging of red blood cells under shear flow, Phys. Rev. Lett. 98 (2007) 188302.

https://doi.org/10.4208/cicp.OA-2016-0164 Published online by Cambridge University Press